Grace Hopper Celebration 2024

Secure Consumption of Open Source Software: Evaluating, Utilizing, and Contributing Safely

Katherine Druckman

Open Source Security Evangelist

Why are we here?

Today we'll cover

- Security challenges in consuming open source software
- Evaluating open source projects through a security lens
- Project health, governance, management, and community
- Tools for securing open source software
- Open source community security efforts: OpenSSF
- How we can contribute to a safer ecosystem

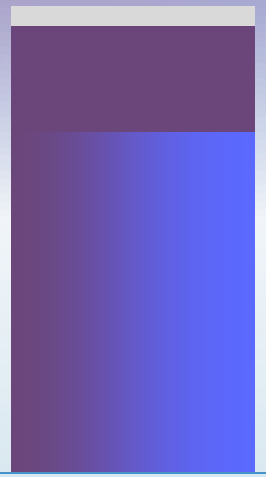
Security challenges

Why is open source security so challenging?

Open source is *everywhere*

96%

of codebases

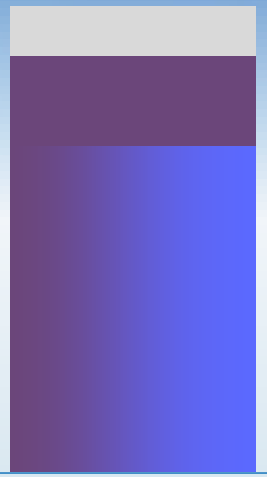


77%

of code within

70–90%

of all software



Source: Synopsis

Source: Linux Foundation

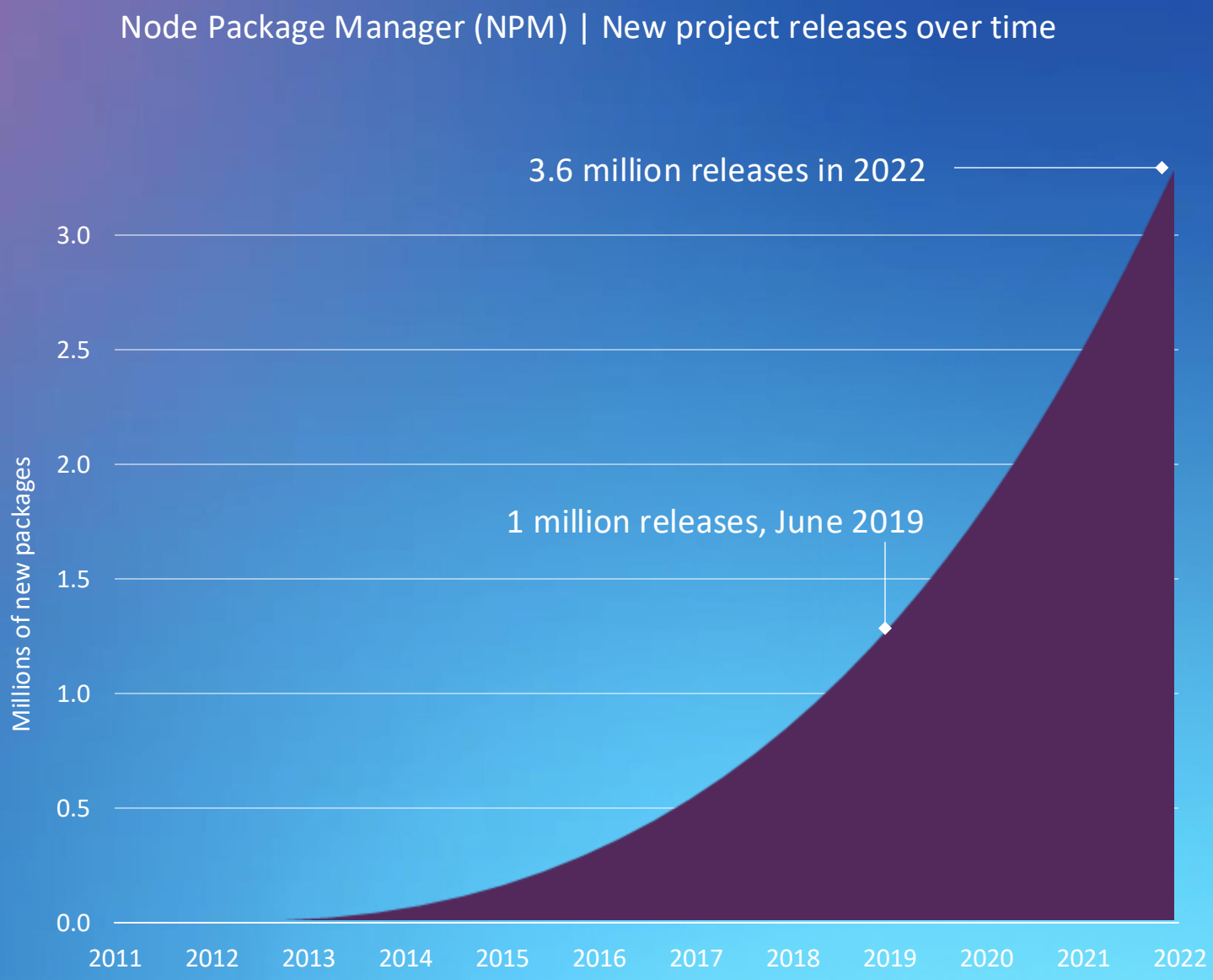
https://www.synopsys.com/software-integrity/resources/analyst-reports/open-source-security-risk-analysis.html

https://www.linuxfoundation.org/blog/blog/a-summary-of-census-ii-open-source-software-application-libraries-the-world-depends-on

Open source is *everywhere*

Yay! We won!

Millions of packages …



https://anchore.com/blog/open-source-is-bigger-than-you-imagine/

Millions of packages … dozens of maintainers

It's not just NPM

npmjs.org
3,639,251 packages
41,815,290 versions
822,231 maintainers
222,295 namespaces
742,169 keywords
237,045,471,901 downloads

proxy.golang.org
1,105,378 packages
9,970,233 versions
449,953 namespaces
70,056 keywords

hub.docker.com
1,001,771 packages
10,844,967 versions
411,451 namespaces
1,713 keywords
334,237,037,105 downloads

nuget.org
624,254 packages
7,566,455 versions
85,940 maintainers
129,237 keywords
514,134,790,590 downloads

pypi.org
542,396 packages
5,603,074 versions
229,691 maintainers
193,748 keywords
29,768,771,520 downloads

repo1.maven.org
499,556 packages
11,361,001 versions
66,310 namespaces
31,287 keywords

Statistics			
Registries: 59	Versions: 97,334,410	Maintainers: 1,440,484	Keywords: 1,521,288
Packages: 8,840,726	Namespaces: 1,328,127	Downloads: 1,811,443,044,372	

Common Vulnerabilities and Exposures (CVE)

CVE is a dictionary of common names for publicly known cybersecurity vulnerabilities, each of which receives a CVE Identifier.

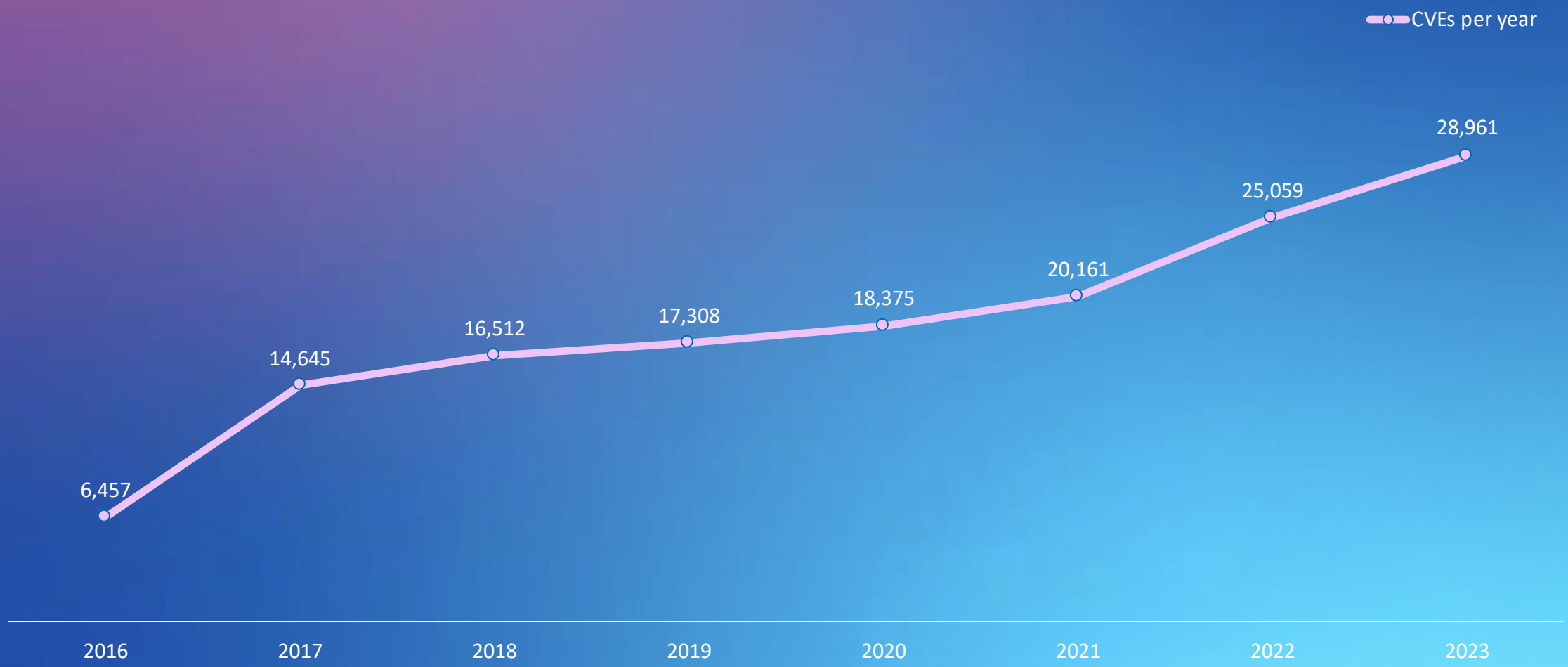
CVE Identifiers make it easy to share data across separate network security databases and tools. Plus, they provide a baseline for evaluating the coverage of an organization's security tools.

- CVE = Common Vulnerabilities and Exposures
- List of "all" publicly known software security vulnerabilities starting in 1999
- MITRE Corporation manages and maintains CVE on behalf of the US National Cybersecurity Division

CVEs: True or false?

A project with no CVEs is more se an a project with many CVEs.

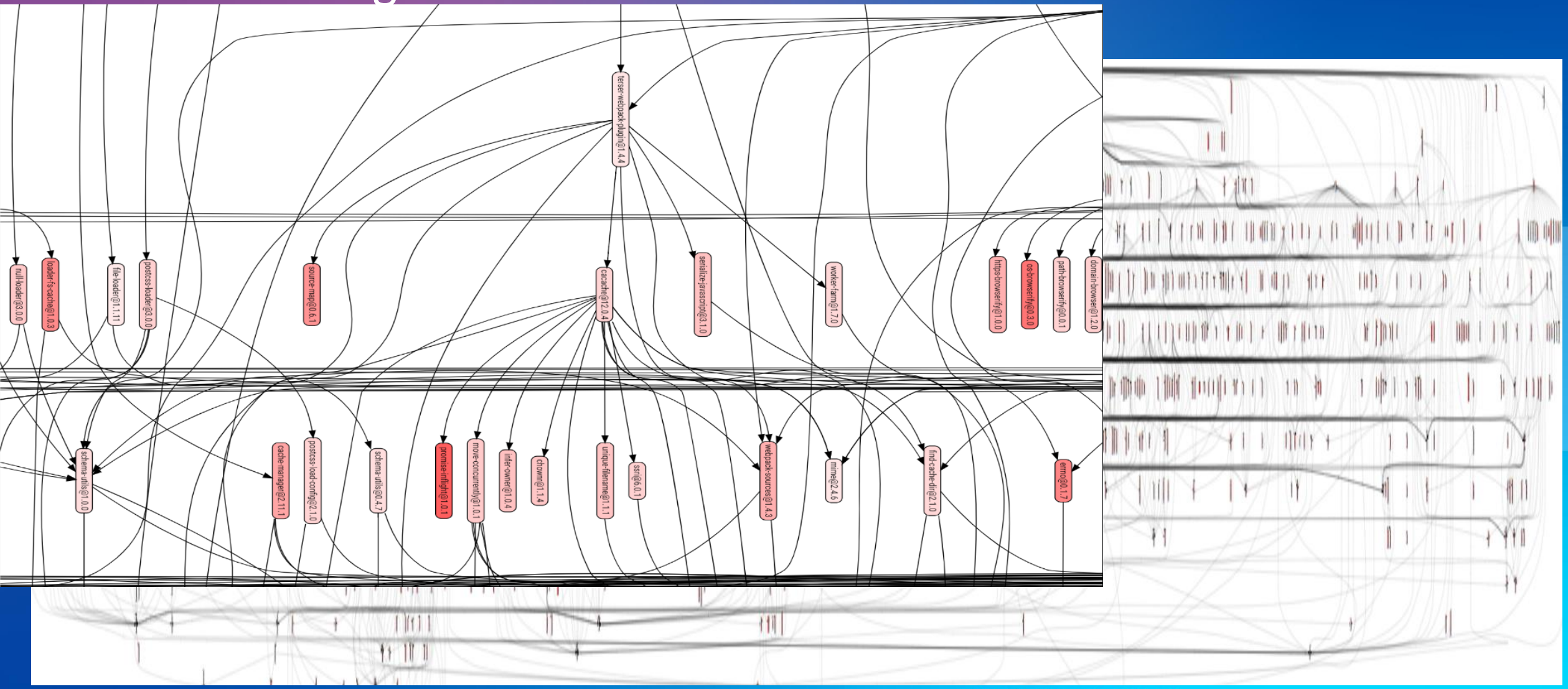
CVEs per year



So many dependencies

- Secondary and tertiary dependencies can get well into the 100s…
- Especially with web applications

How bad it can get



New contributors!

From GitHub:

- 2023 had the largest number of first-time contributors
- 420 million total projects (27% growth YOY)
- 4.5 billion contributions in 2023

Evaluating projects

What does it mean to consume open source software securely?

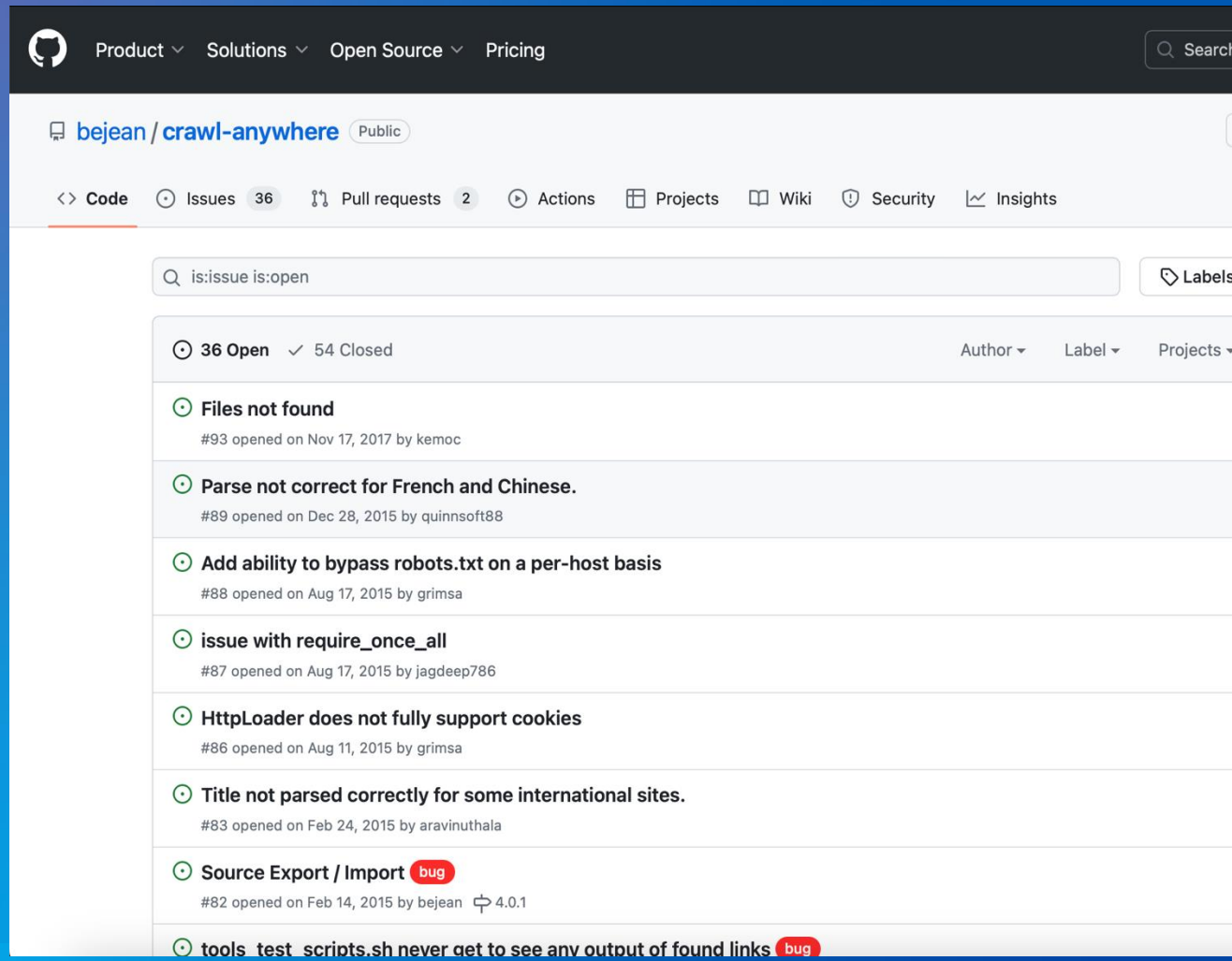
Evaluating open source projects

1. Review basic health—Is it active?
2. Check governance—Is it defined?
3. Review maintenance & releases—Is there a cadence?
4. Explore the community—Are people engaged?
5. Bug reporting—Is there a documented process?

What’s the first thing you would look at when evaluating an open source project to use or include as a dependency?

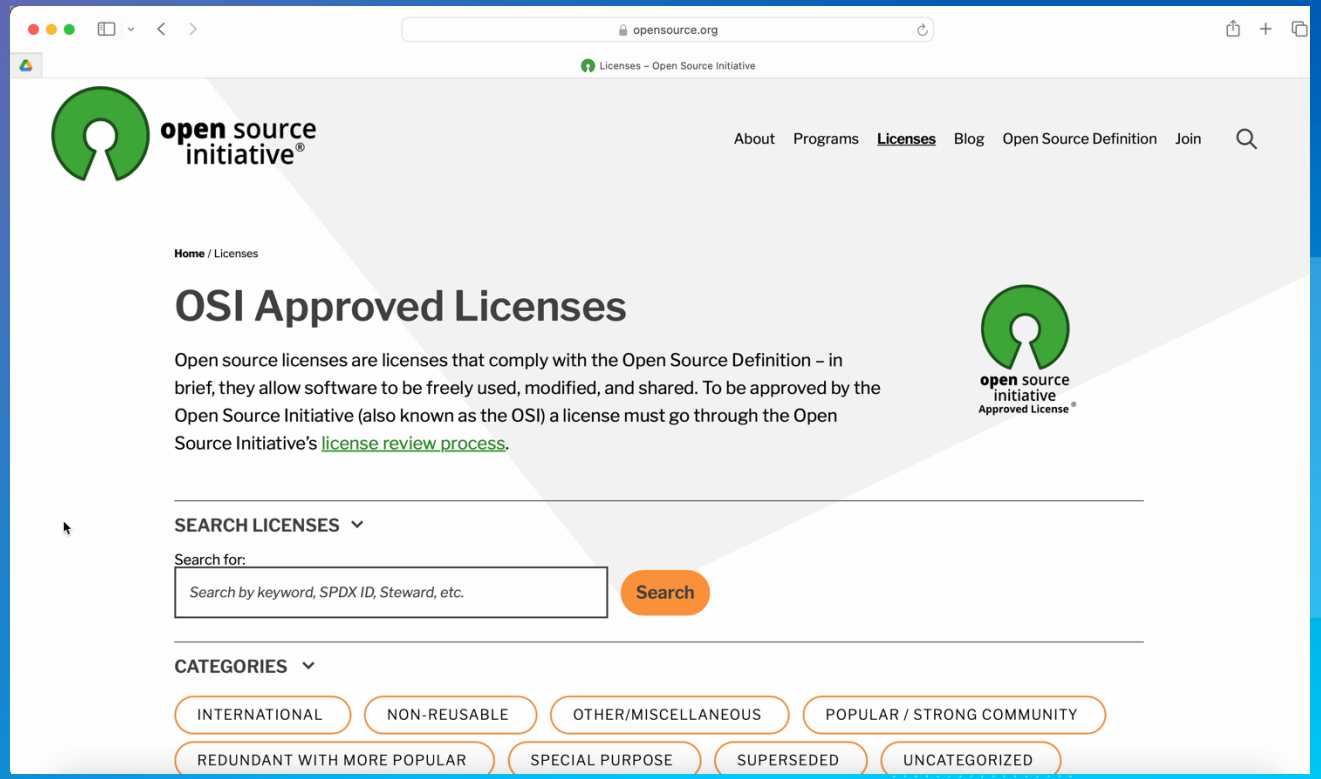
1. Evaluating software: Basic health

- Does the project even have a maintainer anymore?
- When was the last commit?
- Look at the issue queue
  - How active is it?
  - When was the last post?
  - When was the last response to an issue?



2. Evaluating software: Governance

- Clearly defined governance?
  - Clearly stated license? (Hopefully OSI approved)
  - More than one maintainer
  - Maintainers from more than one company or organization
  - How are decisions made?

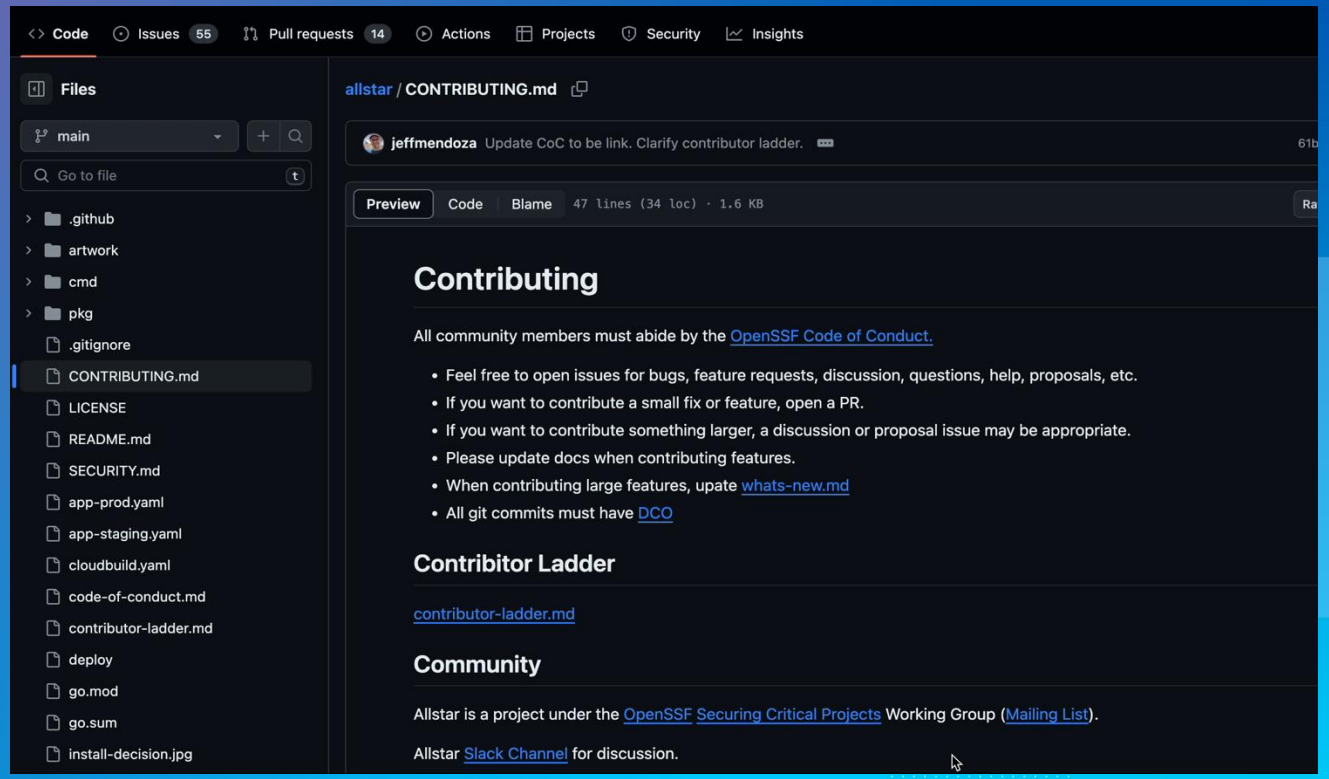


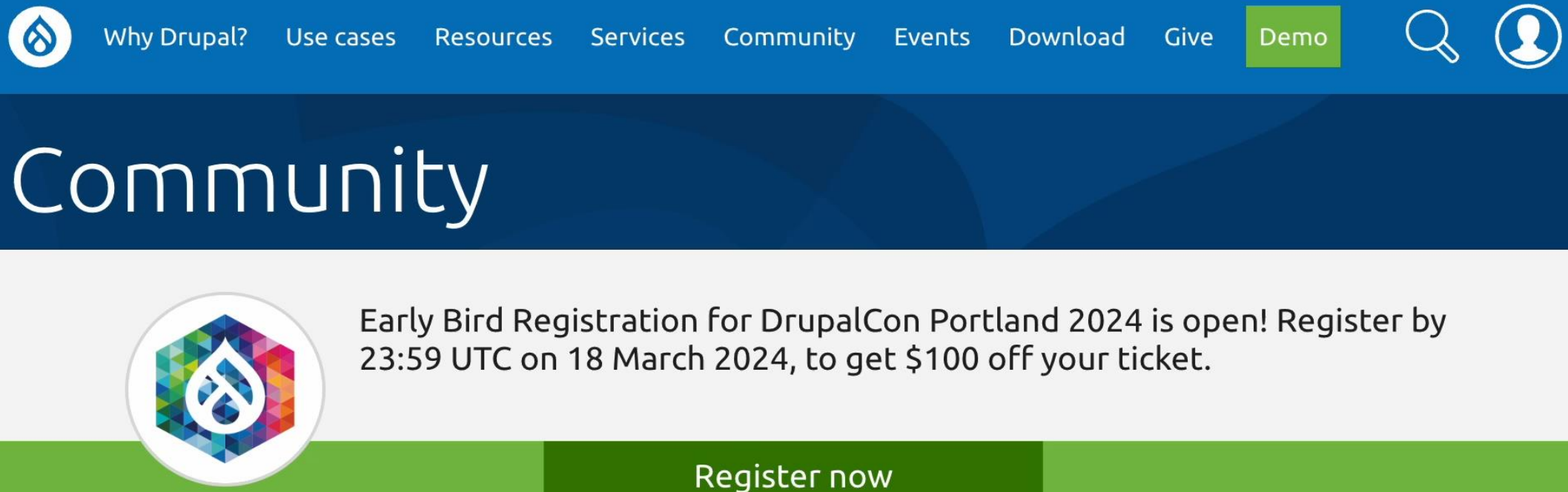
3. Evaluating software: Maintenance and release management

- Has there been substantial activity in the last year?
- Look at the release cadence
  - Is it documented?
  - Regularly occurring?
  - Prompt patch releases to address bugs and security issues?
- Does the project communicate announcements regularly? Does it have a blog?
- Is the latest release a “-alpha” or “-beta,” or does it indicate that it is not yet production-ready?

4. Evaluating software: Community engagement

- Contributor guide?
- Extensively used?
- Is the community working toward security best practices?
  - Automated tests
  - Up-to-date dependencies





Contribution areas

> This guide is always evolving. If you'd like to help improve it, the best starting point is the Contribute to the Contributor Guide page.

The Drupal project has many areas that you can contribute to improving -- it's not

Help improve this page

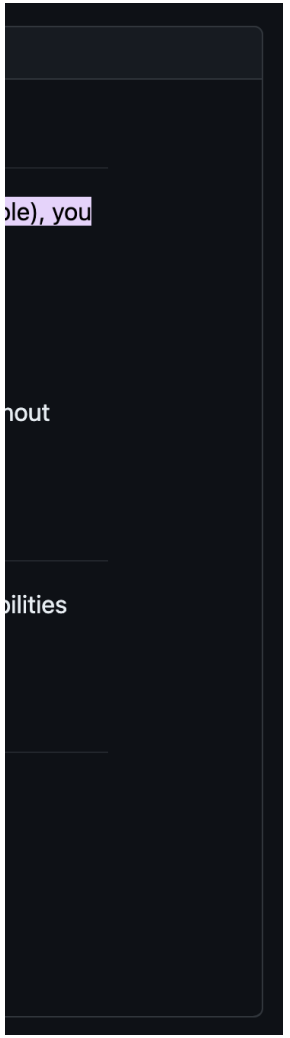
Create an issue describing the problem.

5. Evaluating software: Secure bug reporting

Where do I report security issues?

- If you are here to report any sort of security issue with a site hosted on **WordPress.com**, then please submit a report at the Automattic HackerOne page. If the issue you're trying to report is on WordPress.com and is **not** a security issue, then please use their support forums instead.
- If you're having an issue with your own self-hosted WordPress.org site that is **not** a security issue, then please use the WordPress.org support forums.
- For security issues with WordPress plugins, follow the information on Reporting Plugin Security Issues.
- **For security issues with the self-hosted version of WordPress**, submit a report at the WordPress HackerOne page. Include as much detail as you can. Please **always use HackerOne instead of Core Trac**, even if the vulnerability is only in `trunk`, or a beta/RC release, because there are some sites that run those in production.

In all cases, you should **not** share the details with anyone else until after the fix for the bug has been officially released to the public.



Tools for securing open source software

Beyond basic health

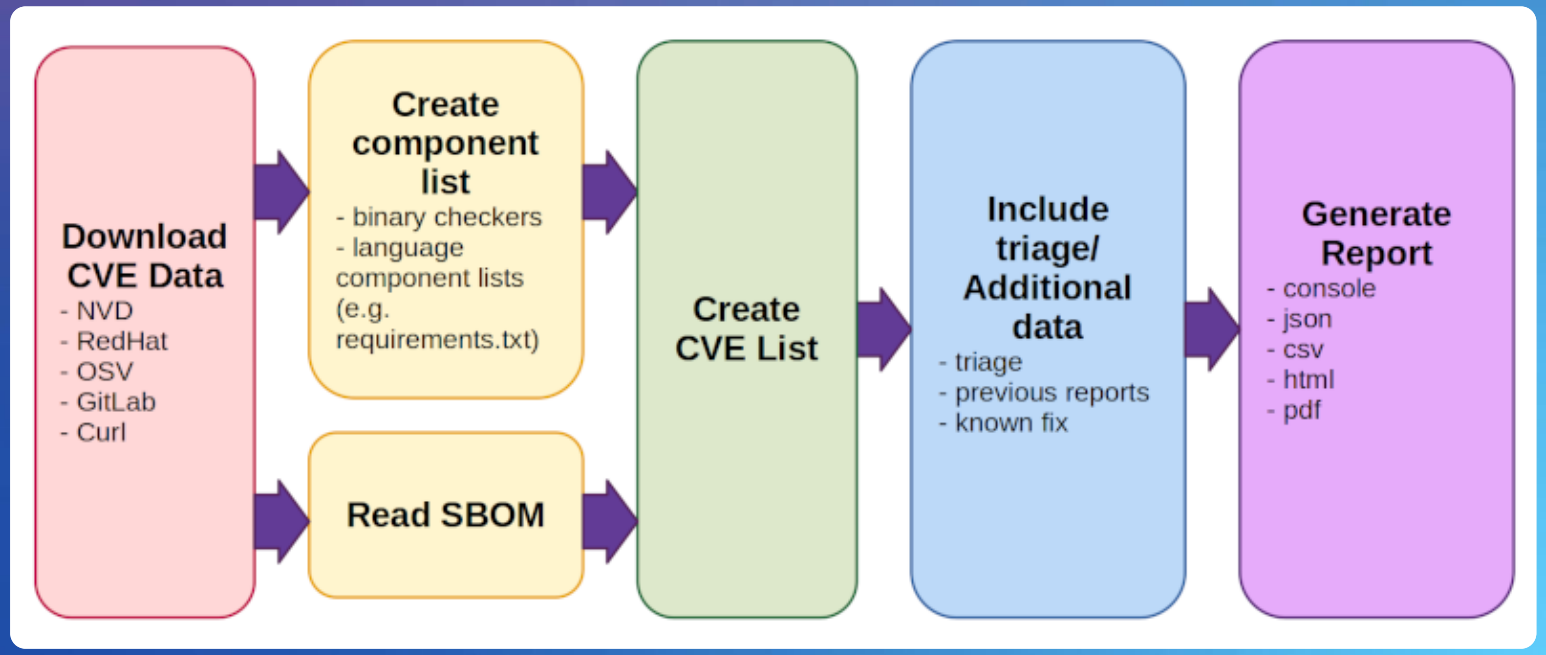
Security tools for open source software

- Intel maintained CVE-bin-tool
- Open Source Security Foundation (OpenSSF):
  - OpenSSF Best Practices Badge
  - Secure Supply Chain Consumption Framework (S2C2F)
  - OpenSSF Scorecard

CVE-bin-tool

The CVE Binary Tool can help you find known vulnerabilities in software by using data from the National Vulnerability Database (NVD) list of Common Vulnerabilities and Exposures (CVEs) as well as known vulnerability data from Redhat, Open Source Vulnerability Database (OSV), Gitlab Advisory Database (GAD), and Curl.

1. A binary scanner—Helps you determine which packages may have been included as part of a piece of software.
2. Tools for scanning known component lists—Such as CSV files, SBOM formats, etc.



As simple as …

pip install cve-bin-tool

cve-bin-tool <directory/file>

Open Source Security Foundation (OpenSSF)

The Open Source Security Foundation (OpenSSF) seeks to make it easy to **develop, maintain, and consume** open source software safely and securely.

This includes fostering collaboration, establishing best practices, and developing innovative solutions for the open source software we all depend on.

Working groups, projects & SIGs

1. INFORM

Vulnerability disclosures
Efficient vulnerability reporting and remediation

I. CVD Guides SIGs
J. OSS-SIRT SIG
K. Open Source Vuln Schema (OSV) project
L. OpenVEX SIG
M. Vuln Autofix SIG

Metrics & metadata
Security metrics/reviews for open source projects

N. Security Insights project
O. Security-Metrics: Risk Dashboard project
P. Security Reviews project
AH. Security Insights Spec project

Securing critical projects
Identification of critical open source projects

U. List of Critical OS Prj, Components & Frameworks SIG
V. Criticality score project
W. Harvard study SIG
X. Package Analysis project
Y. Allstar project

2. EQUIP

Best practices
Identification, awareness, and education of security best practices

A. Secure Software Development Fundamentals courses SIG
B. Security Knowledge Framework (SKF) project
C. OpenSSF Best Practices Badge project
D. OpenSSF Scorecard project
E. Common Requirements Enumeration (CRE) project
F. Concise & Best Practices Guides SIGs
G. Education SIG
H. Memory Safety SIG
AG. The Security Toolbelt SIG

Security tooling
State of the art security tools

Q. SBOM Everywhere SIG
R. OSS Fuzzing SIG
AI. SBOMit project
Protobom project

AI/ML security
AI/ML security at the Intersection of Artificial Intelligence and Cybersecurity

DevRel
Develop Use Cases and help others learn about security

Supply chain integrity
Ensuring the provenance of open source code

S. Supply-chain Levels for Software Artifacts (SLSA) project
T. Secure Supply Chain Consumpt Framework (S2C2F) project
AJ. Gittuf project
AK. GUAC project

Diversity, equity & inclusion
Increase representation and strengthen the overall effectiveness of the cybersecurity workforce

3. ENGAGE

End users
Voice of public & private sector organizations that primarily consume open source

Z. Threat Modeling SIG

Securing software repositories
Collaboration between repository operators

AB. RSTUF project

Projects
Category-leading software initiatives

AD. Alpha-Omega
AE. Sigstore
AF. Core Toolchain Infrastructure (CTI)

OpenSSF Best Practices Badge

README LGPL-3.0 license Security

Gramine Library OS with I

docs passing openssf best practices passing

A Linux-compatible Library for Multi-Process A

What is Gramine?

Gramine (formerly called *Graphen* lightweigh
host requirements. Gramine can run cations in
a complete OS in a virtual machine -- luding gue
process migration.

Gramine supports native, unmodified Linux binaries
SGX enclaves on Linux platforms.

In untrusted cloud and edge deployments, there is
infrastructure. Gramine supports this "lift and shift"
Confidential Computing with Intel SGX. Gramine ca
minimal porting effort.

Documentation

?	Met / Unmet / ?	The project MUST have a documented roadmap that describes what the project intends to do and not do for at least the next year. (URL required) [documentation_roadmap] Show details
?	Met / Unmet / N/A / ?	The project MUST include documentation of the architecture (aka high-level design) of the software produced by the project. If the project does not produce software, select "not applicable" (N/A). (URL required) [documentation_architecture] Show details
?	Met / Unmet / N/A / ?	The project MUST document what the user can and cannot expect in terms of security from the software produced by the project (its "security requirements"). (URL required) [documentation_security] Show details
✓	Met / Unmet / N/A / ?	The project MUST provide a "quick start" guide for new users to help them quickly do something with the software. (URL required) [documentation_quick_start] Show details

https://www.libreoffice.org/get-help/documentation/

OpenSSF Scorecard

- What is it?
  - Quick, easy project assessment via list of automated checks for best practices
- What does it help protect me from?
  - Malicious maintainers and packages
  - Poorly maintained projects
  - Compromised build systems and/or code
- How do I use it?
  - Command line interface (CLI)
  - GitHub Action

README | Code of conduct | Apache-2.0 license | Security

OpenSSF Scorecard

openssf scorecard 9.6 | openssf best practices passing | build passing | CodeQL passing | reference | go report A+
codecov 75% | SLSA level 3 | slack openssf/scorecard

Overview

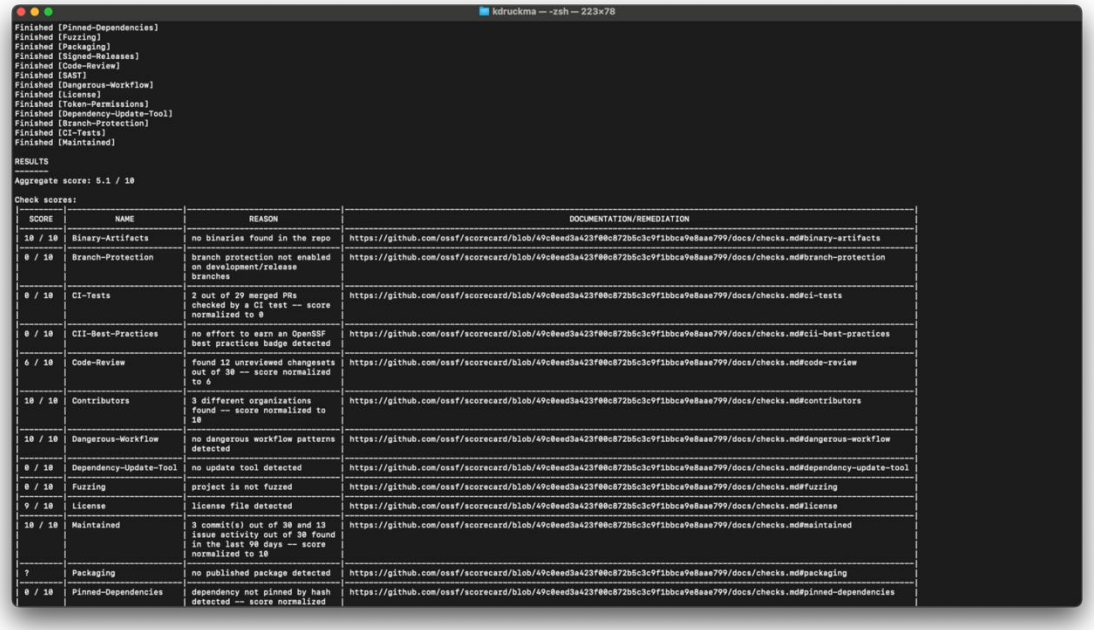
- What Is Scorecard?
- Prominent Scorecard Users
- View a Project's Score
- Scorecard's Public Data

Using Scorecard

- Scorecard GitHub Action
- Scorecard REST API
- Scorecard Badges
- Scorecard Command Line Interface
  - Prerequisites
  - Installation
  - Authentication
  - Basic Usage

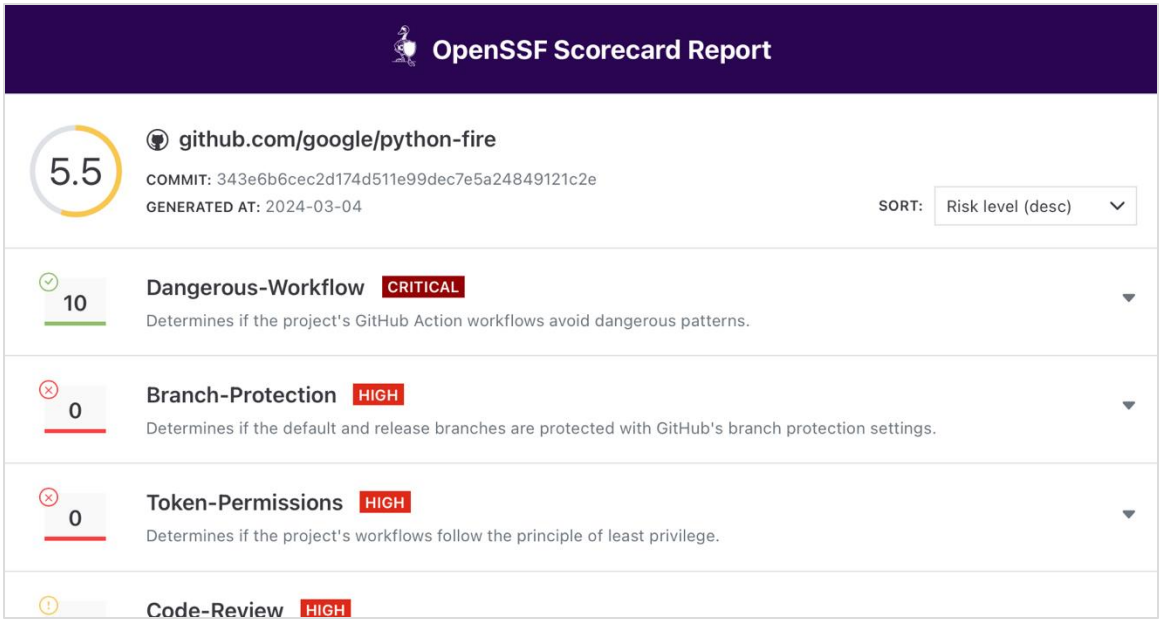
OpenSSF Scorecard

Score in terminal …



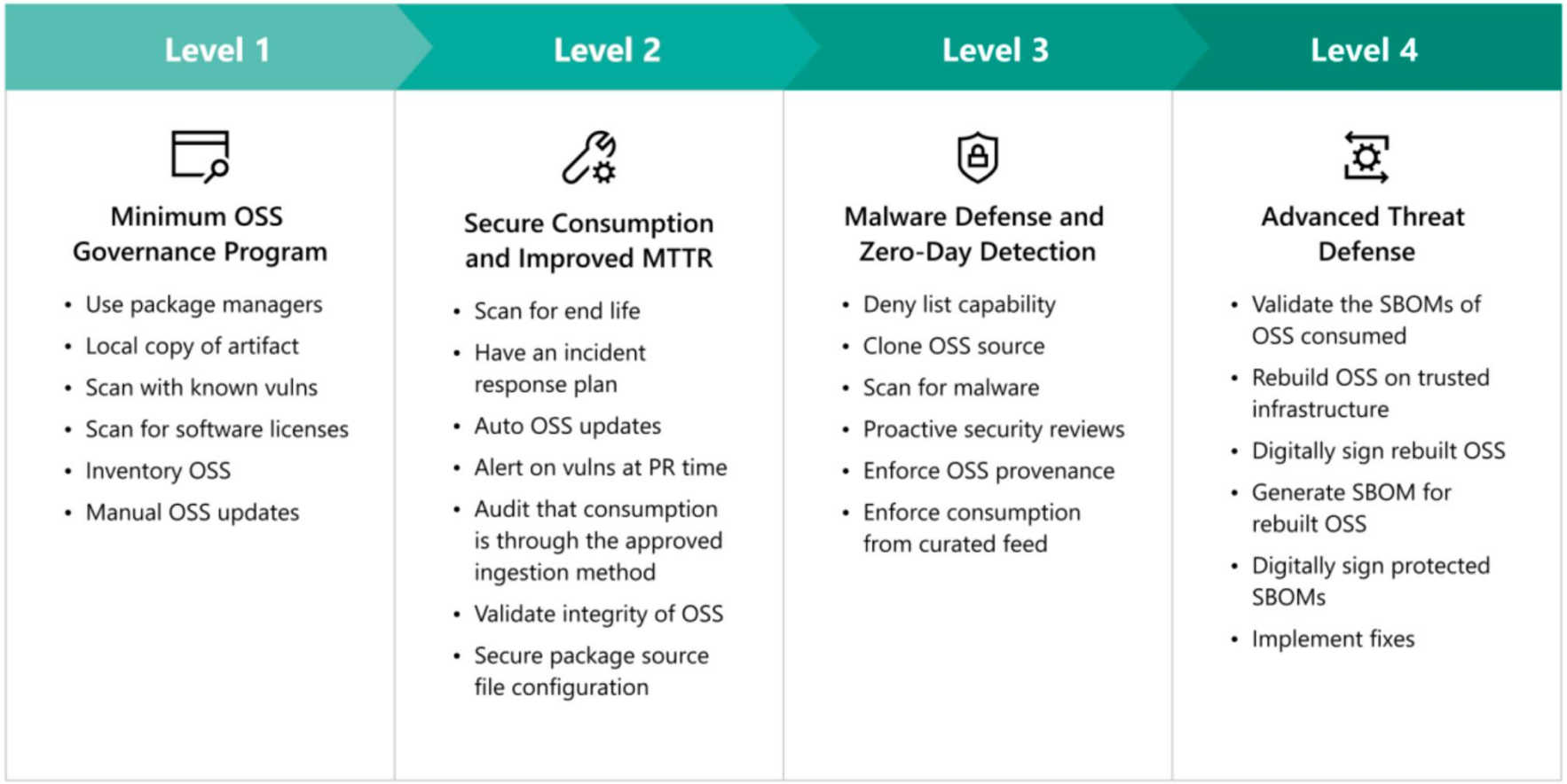
```
kdruckma — -zsh — 223×78
Finished [Pinned-Dependencies]
Finished [Fuzzing]
Finished [Packaging]
Finished [Signed-Releases]
Finished [Code-Review]
Finished [SAST]
Finished [Dangerous-Workflow]
Finished [License]
Finished [Token-Permissions]
Finished [Dependency-Update-Tool]
Finished [Branch-Protection]
Finished [CI-Tests]
Finished [Maintained]

RESULTS
-------
Aggregate score: 5.1 / 10

Check scores:
| SCORE | NAME | REASON | DOCUMENTATION/REMEDIATION |
| 10 / 10 | Binary-Artifacts | no binaries found in the repo | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#binary-artifacts |
| 0 / 10 | Branch-Protection | branch protection not enabled on development/release branches | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#branch-protection |
| 0 / 10 | CI-Tests | 2 out of 29 merged PRs checked by a CI test -- score normalized to 0 | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#ci-tests |
| 0 / 10 | CII-Best-Practices | no effort to earn an OpenSSF best practices badge detected | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#cii-best-practices |
| 6 / 10 | Code-Review | found 12 unreviewed changesets out of 30 -- score normalized to 6 | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#code-review |
| 10 / 10 | Contributors | 3 different organizations found -- score normalized to 10 | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#contributors |
| 10 / 10 | Dangerous-Workflow | no dangerous workflow patterns detected | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#dangerous-workflow |
| 0 / 10 | Dependency-Update-Tool | no update tool detected | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#dependency-update-tool |
| 0 / 10 | Fuzzing | project is not fuzzed | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#fuzzing |
| 9 / 10 | License | license file detected | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#license |
| 10 / 10 | Maintained | 3 commit(s) out of 30 and 13 issue activity out of 30 found in the last 90 days -- score normalized to 10 | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#maintained |
| ? | Packaging | no published package detected | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#packaging |
| 0 / 10 | Pinned-Dependencies | dependency not pinned by hash detected -- score normalized | https://github.com/ossf/scorecard/blob/49c0eed3a423f00c872b5c3c9f1bbca9e8aae799/docs/checks.md#pinned-dependencies |
```

… or via browser



Secure Supply Chain Consumption Framework (S2C2F)

The S2C2F project works to further develop and improve the S2C2F guide, which outlines how to securely consume open source software (OSS) dependencies.



Putting the tools to work

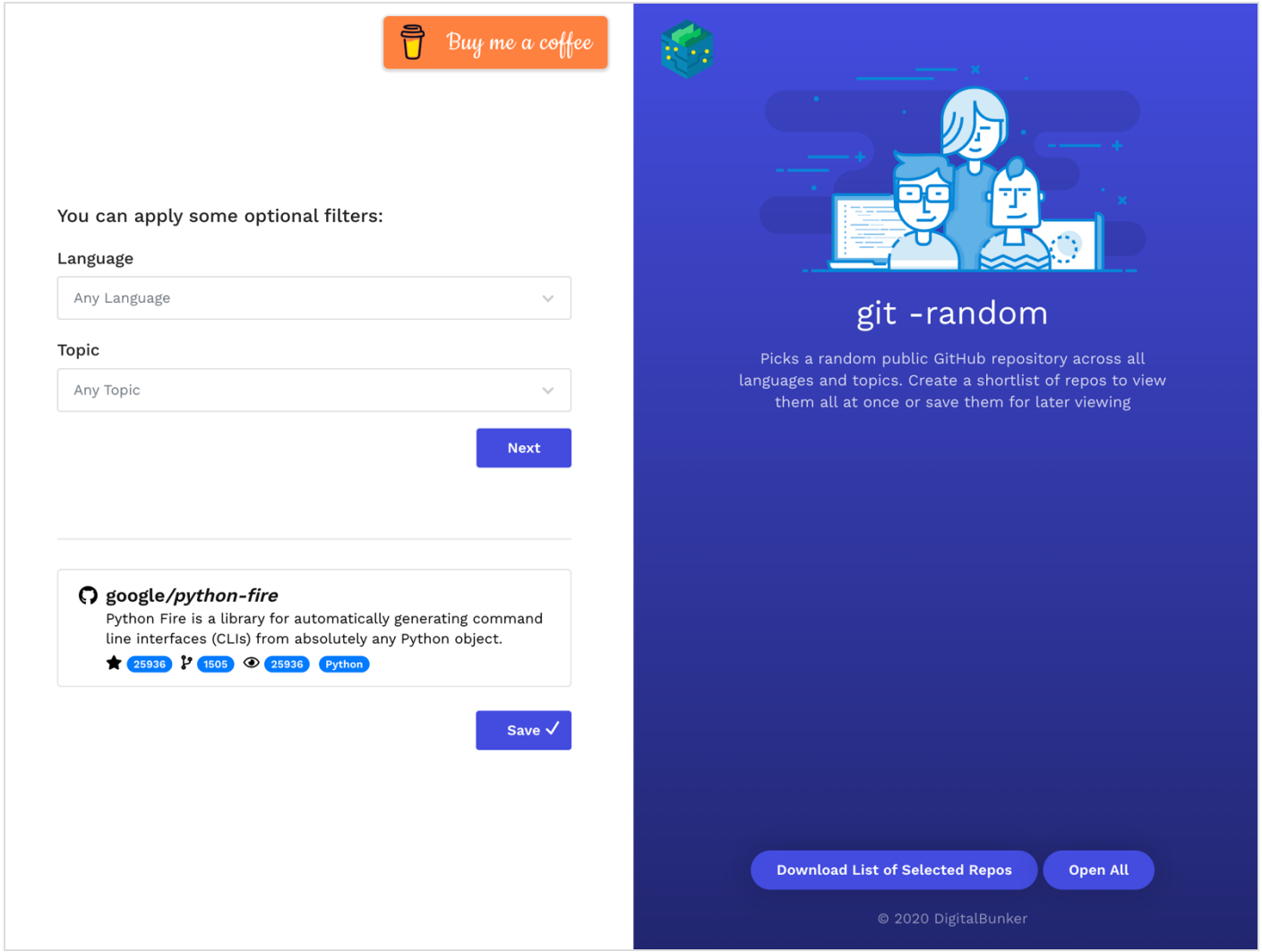
Let’s evaluate some software

Evaluating open source projects

1. Review basic health—Is it active?
2. Check governance—Is it defined?
3. Review maintenance & releases—Is there a cadence?
4. Explore the community—Are people engaged?
5. Bug reporting—Is there a documented process?
6. Run OpenSSF Scorecard

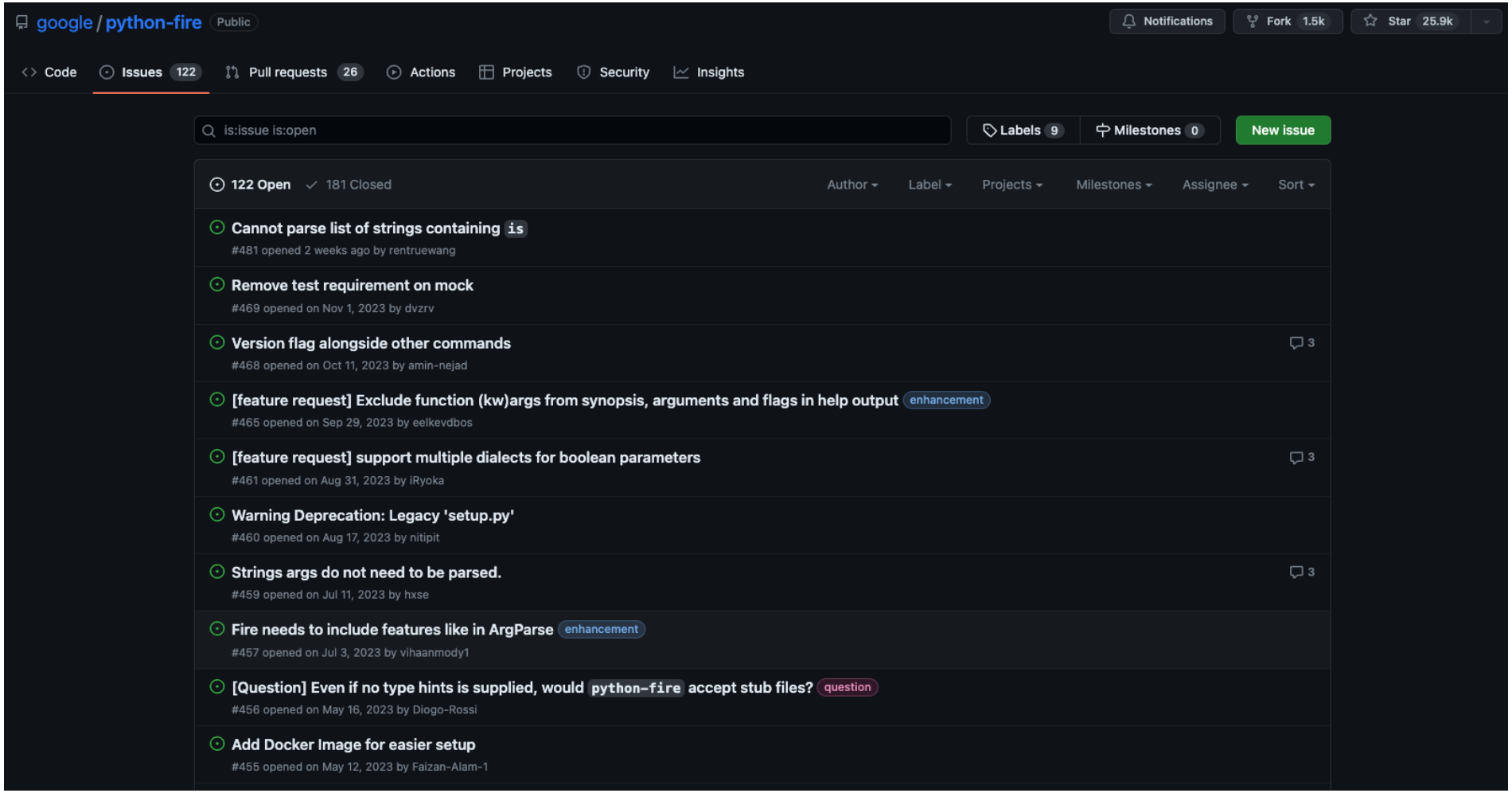
OpenSSF projects and tools

Grab a random repo

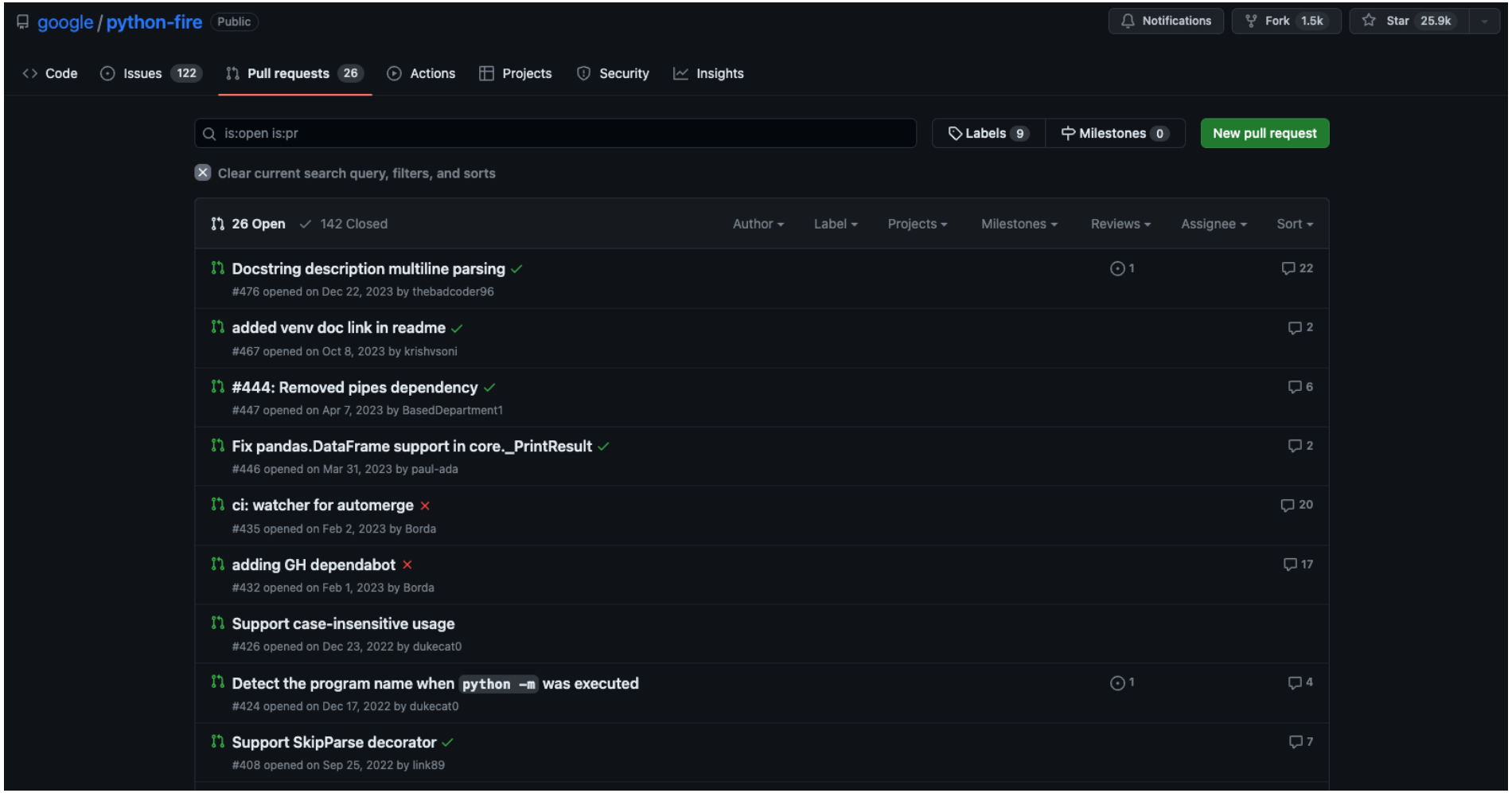


Basic health check: Looks promising!

Issue queue


google / python-fire Public
Notifications
Fork 1.5k
Star 25.9k
Code
Issues 122
Pull requests 26
Actions
Projects
Security
Insights
is:issue is:open
Labels 9
Milestones 0
New issue
122 Open
181 Closed
Author
Label
Projects
Milestones
Assignee
Sort
Cannot parse list of strings containing is
#481 opened 2 weeks ago by rentruewang
Remove test requirement on mock
#469 opened on Nov 1, 2023 by dvzrv
Version flag alongside other commands
3
#468 opened on Oct 11, 2023 by amin-nejad
[feature request] Exclude function (kw)args from synopsis, arguments and flags in help output enhancement
#465 opened on Sep 29, 2023 by eelkevdbos
[feature request] support multiple dialects for boolean parameters
3
#461 opened on Aug 31, 2023 by iRyoka
Warning Deprecation: Legacy 'setup.py'
#460 opened on Aug 17, 2023 by nitipit
Strings args do not need to be parsed.
3
#459 opened on Jul 11, 2023 by hxse
Fire needs to include features like in ArgParse enhancement
#457 opened on Jul 3, 2023 by vihaanmody1
[Question] Even if no type hints is supplied, would python-fire accept stub files? question
#456 opened on May 16, 2023 by Diogo-Rossi
Add Docker Image for easier setup
#455 opened on May 12, 2023 by Faizan-Alam-1

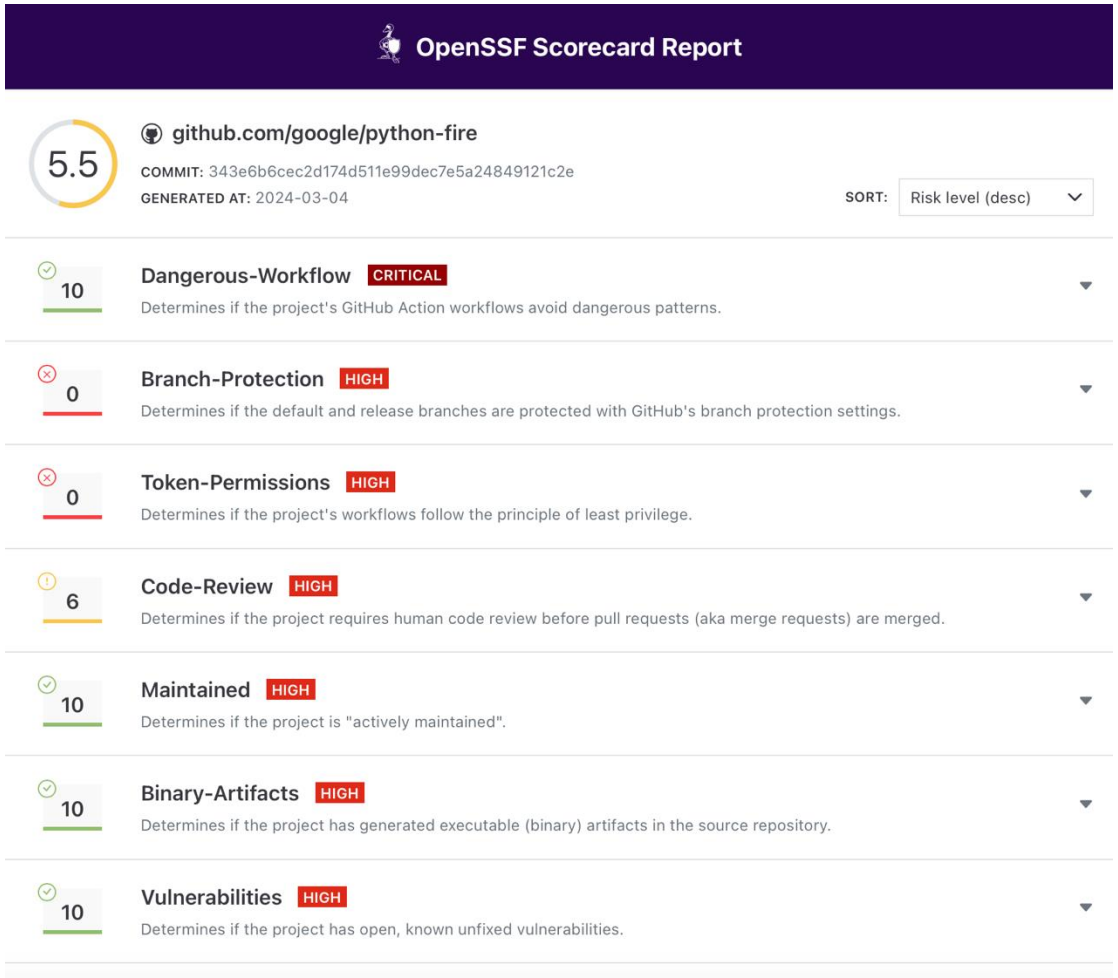
Pull requests

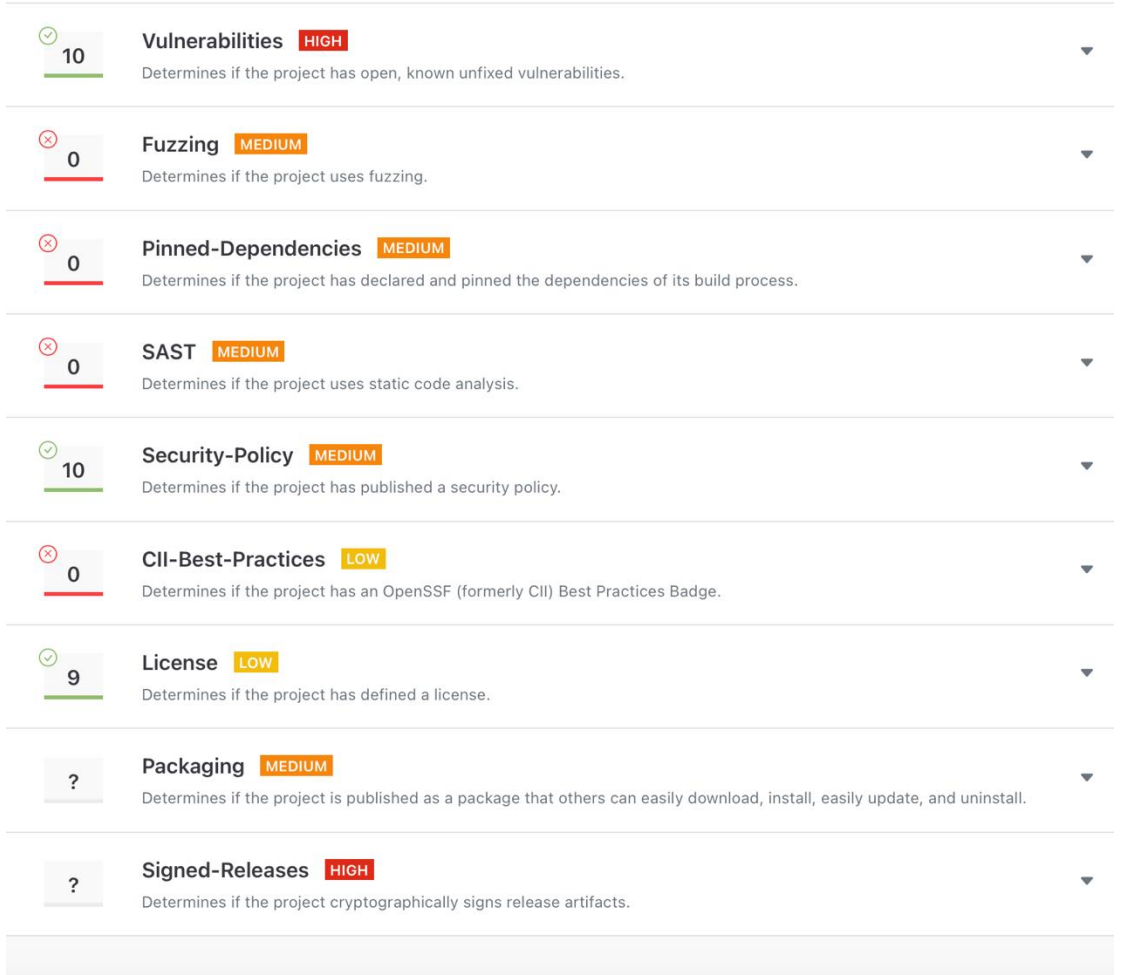


OpenSSF Scorecard: Manual CLI scan—Terminal

scorecard --repo github.com/google/python-fire

OpenSSF Scorecard: Manual CLI scan—Browser





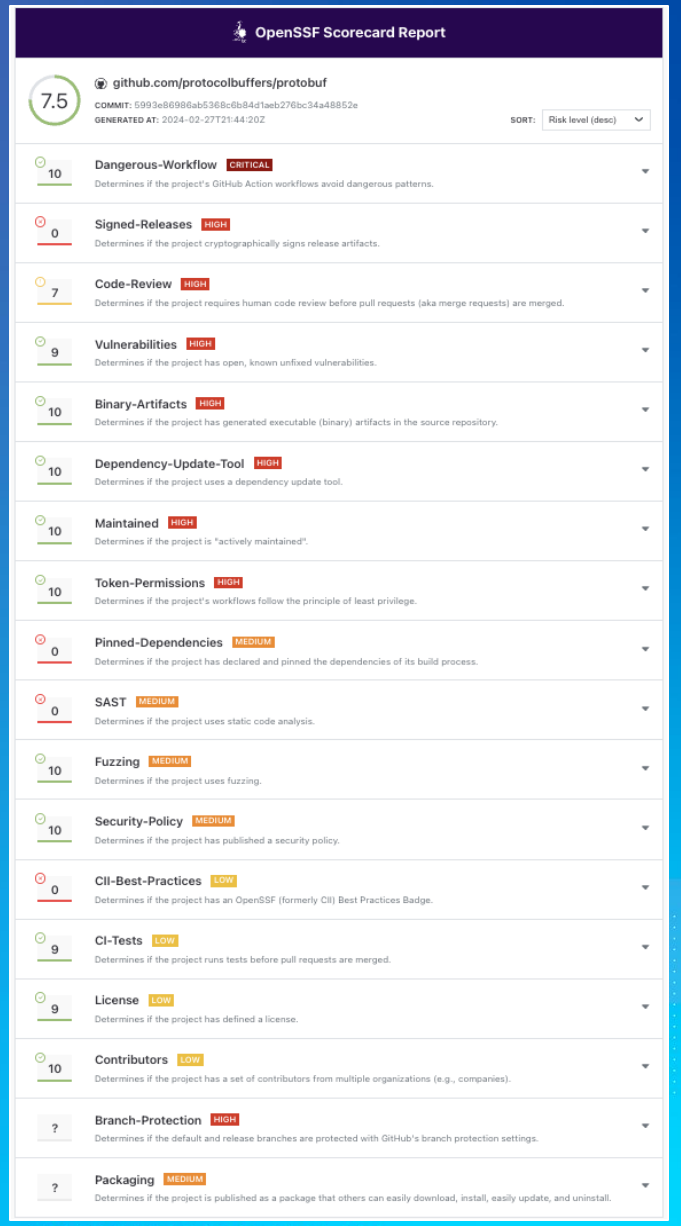
OpenSSF web report: Protocol buffers

The good:

- No dangerous workflows!
- Maintained!
- Security policy!
- Even fuzzing!

The less good:

- No signed releases
- Static analysis
- Branch protection unknown



Get involved in the OpenSSF

Be your own hero!

openssf.org

The best way to influence an OSS project direction is to get involved.

Taking ownership

Developers don’t owe you anything

Scan for tools, sources, and resources

Visit the presentation's GitHub page

- Intel and OpenSSF tools
- Guides and community resources
- Links to articles and source material
- A PDF of this presentation

Where to find me

- Twitter/X: @katherined
- Fediverse: @katherined@reality2.social
- LinkedIn: katherinedruckman

Notices and disclaimers

Intel is committed to respecting human rights and avoiding complicity in human rights abuses. See Intel's Global Human Rights Principles. Intel® products and software are intended only to be used in applications that do not cause or contribute to a violation of an internationally recognized human right.

Intel® technologies may require enabled hardware, software, or service activation. No product or component can be absolutely secure. Your costs and results may vary.